# Database Technology

## Topic 5: SQL

Olaf Hartig

olaf.hartig@liu.se

Make sure you have a sheet of paper and a pen.

You may need them for some of the exercises today.

LINKÖPING UNIVERSITY

# Outline

- SQL data model
- SQL as a data definition language
- SQL queries
  - simple queries
  - join queries
  - set operations
  - subqueries
  - grouping + aggregation
- SQL data manipulation operations
- SQL views

# SQL Data Model

# SQL Data Model

- Based on the relational data model
- Terminology:

| Relational Model | SQL |
|---|---|
| relation | table |
| tuple | row |
| attribute | column |

- In contrast to the relational model, SQL allows duplicate rows in table and in query results

# Question

Go to **www.menti.com** and use the code 8610 7857

Why does SQL allow duplicate tuples
in a table or in a query result?

# SQL Data Model

- Based on the relational data model
- Terminology:

| Relational Model | SQL |
|---|---|
| relation | table |
| tuple | row |
| attribute | column |

- In contrast to the relational model, SQL allows duplicate rows in table and in query results
  - Removing duplicates is expensive
  - User may want information about duplicates
  - Aggregation operators (e.g., sum)

# SQL DDL

# Exercise

- Consider the following two tables

Instructor

| ID | Name | Office |
|---|---|---|
| 4 | Jennifer | B308 |
| 35 | Paul | B311 |
| 12 | Kim | E112 |

Course

| CourseID | Year | Instructor |
|---|---|---|
| cid444 | 2012 | 35 |
| cid598 | 2013 | 4 |
| cid444 | 2013 | 35 |

- Assume that the *Instructor* table has already been created; provide the SQL statement to create the *Course* table, including all of its integrity constraints.

# Exercise

- Consider the following two tables

Instructor

| ID | Name | Office |
|---|---|---|
| 4 | Jennifer | B308 |
| 35 | Paul | B311 |
| 12 | Kim | E112 |

Course

| CourseID | Year | Instructor |
|---|---|---|
| cid444 | 2012 | 35 |
| cid598 | 2013 | 4 |
| cid444 | 2013 | 35 |

- Assume that the *Instructor* table has already been created; provide the SQL statement to create the *Course* table, including all of its integrity constraints.

```
CREATE TABLE Course (
  CourseID CHAR(6),
  Year INTEGER,
  Instructor INTEGER,

  PRIMARY KEY (CourseID, Year),
  FOREIGN KEY (Instructor) REFERENCES Instructor(ID)
);
```

# SQL Queries

## Simple Queries

# Quiz

Consider the following two tables

Instructor

| ID | Name | Office |
|---|---|---|
| 4 | Jennifer | B308 |
| 35 | Paul | B311 |
| 12 | Kim | E112 |

Course

| CourseID | Year | Instructor |
|---|---|---|
| cid444 | 2012 | 35 |
| cid598 | 2013 | 4 |
| cid444 | 2013 | 35 |

What is the result of the following query?

**SELECT** CourseID **FROM** Course **WHERE** Instructor = 35;

**1:**

| CourseID |
|---|
| cid444 |
| cid598 |
| cid444 |

**2:**

| CourseID | Instructor |
|---|---|
| cid444 | 35 |
| cid444 | 35 |

**3:**

| CourseID |
|---|
| cid444 |

**4:**

| CourseID |
|---|
| cid444 |
| cid444 |

# SQL Queries

## Join Queries

# Quiz

Consider the following two tables

Instructor

| ID | Name | Office |
|---|---|---|
| 4 | Jennifer | B308 |
| 35 | Paul | B311 |
| 12 | Kim | E112 |

Course

| CourseID | Year | Instructor |
|---|---|---|
| cid444 | 2012 | 35 |
| cid598 | 2013 | 4 |
| cid444 | 2013 | 35 |

How many rows do we have in the result of the following query?

**SELECT** CourseID **FROM** Course, Instructor **WHERE** Year = 2013;

1) 2 rows
2) 4 rows
3) 6 rows
4) 8 rows

# Quiz

Consider the following two tables

Instructor

| ID | Name | Office |
|---|---|---|
| 4 | Jennifer | B308 |
| 35 | Paul | B311 |
| 12 | Kim | E112 |

Course

| CourseID | Year | Instructor |
|---|---|---|
| cid444 | 2012 | 35 |
| cid598 | 2013 | 4 |
| cid444 | 2013 | 35 |

How many rows do we have in the result of the following query?

**SELECT** Name, CourseID
**FROM** Instructor **LEFT OUTER JOIN** Course **ON** ID = Instructor;

1) 2 rows
2) 3 rows
3) 4 rows
4) 6 rows

# SQL Queries

## Set Operations

# Quiz

Consider the following two tables

Instructor

| ID | Name | Office |
|---|---|---|
| 4 | Jennifer | B308 |
| 35 | Paul | B311 |
| 12 | Kim | E112 |

Course

| CourseID | Year | Instructor |
|---|---|---|
| cid444 | 2012 | 35 |
| cid598 | 2013 | 4 |
| cid444 | 2013 | 35 |

How many rows do we have in the result of the following query?

```
SELECT ID FROM Instructor
UNION
SELECT Instructor FROM Course;
```

1) 3 rows
2) 5 rows
3) 6 rows
4) none, we get an error message

# Exercise

Consider the following two tables

Instructor

| ID | Name | Office |
|---|---|---|
| 4 | Jennifer | B308 |
| 35 | Paul | B311 |
| 12 | Kim | E112 |

Course

| CourseID | Year | Instructor |
|---|---|---|
| cid444 | 2012 | 35 |
| cid598 | 2013 | 4 |
| cid444 | 2013 | 35 |

Write an SQL query the returns all instructor IDs of instructors who are not assigned to any course.

Hence, for the example data above, the query result should be:

| ID |
|---|
| 12 |

# SQL Queries

## Subqueries

# Quiz

Consider the following two tables

Instructor

| ID | Name | Office |
|---|---|---|
| 4 | Jennifer | B308 |
| 35 | Paul | B311 |
| 12 | Kim | E112 |

Course

| CourseID | Year | Instructor |
|---|---|---|
| cid444 | 2012 | 35 |
| cid598 | 2013 | 4 |
| cid444 | 2013 | 35 |

Which names are in the result of the following query?

```
SELECT Name
FROM Instructor
WHERE ID NOT IN ( SELECT Instructor FROM Course );
```

1) Paul, Kim
2) Jennifer, Paul
3) Kim
4) Jennifer

# Quiz

Consider the following two tables

Instructor

| ID | Name | Office |
|---|---|---|
| 4 | Jennifer | B308 |
| 35 | Paul | B311 |
| 12 | Kim | E112 |

Course

| CourseID | Year | Instructor |
|---|---|---|
| cid444 | 2012 | 35 |
| cid598 | 2013 | 4 |
| cid444 | 2013 | 35 |

Is the subquery in the following query a correlated subquery?

```
SELECT Name
FROM Instructor
WHERE ID NOT IN ( SELECT Instructor FROM Course );
```

1) yes
2) no

# Correlated Subqueries

Consider the following two tables

Instructor

| ID | Name | Office |
|---|---|---|
| 4 | Jennifer | B308 |
| 35 | Paul | B311 |
| 12 | Kim | E112 |

Course

| CourseID | Year | Instructor |
|---|---|---|
| cid444 | 2012 | 35 |
| cid598 | 2013 | 4 |
| cid444 | 2013 | 35 |

Here is a semantically equivalent query with a correlated subquery:

```
SELECT Name
FROM Instructor
WHERE NOT EXISTS ( SELECT *
                   FROM Course
                   WHERE Instructor = ID );
```

# Queries with Set Operations

Consider the following two tables

Instructor

| ID | Name | Office |
|---|---|---|
| 4 | Jennifer | B308 |
| 35 | Paul | B311 |
| 12 | Kim | E112 |

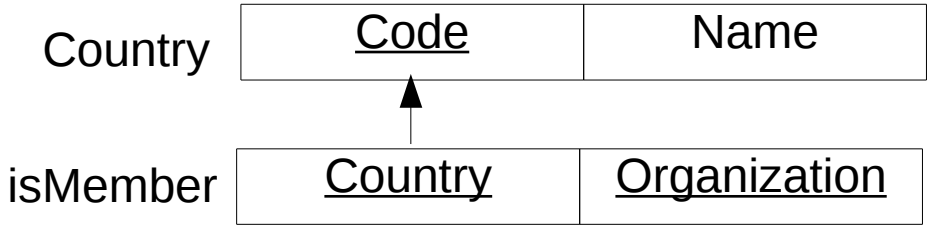
Course

| CourseID | Year | Instructor |
|---|---|---|
| cid444 | 2012 | 35 |
| cid598 | 2013 | 4 |
| cid444 | 2013 | 35 |

Here is another semantically equivalent query with a set operation:

```
SELECT Name
FROM Instructor
WHERE ID IN ( SELECT ID FROM Instructor
              EXCEPT
              SELECT Instructor FROM Course );
```

# Quiz




- Are the following two SQL queries semantically equivalent? (i.e., are their respective results equivalent to one another, for all possible instances of the two tables)

1) yes 2) no

```
SELECT Name
FROM Country
WHERE Code IN ( SELECT Country
                FROM IsMember
                WHERE Organization = 'EU' );
```

---

```
SELECT Name
FROM Country, IsMember
WHERE Code = Country
  AND Organization = 'EU';
```

# Quiz

Which of the following statements *is correct*?

1) The SELECT clause and the FROM clause are the only parts of an SQL query that can influence the number of columns in the query result.
2) Every table can be mentioned only once in the FROM clause.
3) The join condition of a join query must be specified in the WHERE clause.
4) There is no way the SELECT clause can have an effect on the number of rows of the query result.

# SQL Queries

## Grouping and Aggregation

# Quiz

Consider the table:

Instructor

| ID | Name | Office | Salary |
|---|---|---|---|
| 4 | Jennifer | B308 | 40000 |
| 35 | Paul | B311 | 20000 |
| 12 | Kim | E112 | *NULL* |

Consider the following query:

```
SELECT  COUNT(*), COUNT(Salary), AVG(Salary)
FROM Instructor;
```

The result of this query consists of a single row. Which of the following rows would that be for the Instructor table given above?

1) [ 3, 3, 30000 ]
2) [ 3, 3, 20000 ]
3) [ 3, 2, 30000 ]
4) [ 3, 2, NULL ]

# Quiz

Given the *Course2* table shown here:

… how many rows do we have in the result of the following query?

```
SELECT  CourseID
FROM Course2
WHERE Year > 2012
GROUP BY CourseID
HAVING COUNT(*) > 2;
```

1) 1 row
2) 2 rows
3) 3 rows
4) 4 rows

Course2

| CourseID | Year | Instructor |
|---|---|---|
| cid444 | 2012 | 35 |
| cid444 | 2013 | 35 |
| cid444 | 2014 | 35 |
| cid610 | 2014 | 12 |
| cid610 | 2015 | 4 |
| cid610 | 2016 | 12 |
| cid598 | 2013 | 4 |
| cid598 | 2014 | 4 |
| cid777 | 2014 | 35 |

# Group Activity (Bonus)

Given the *Course2* table shown here write an SQL query such that

i) the query contains **no WHERE clause** (SELECT, FROM, GROUP BY, and HAVING are allowed), and

ii) the query results consists of **exactly 7 rows**

Course2

| CourseID | Year | Instructor |
|---|---|---|
| cid444 | 2012 | 35 |
| cid444 | 2013 | 35 |
| cid444 | 2014 | 35 |
| cid610 | 2014 | 12 |
| cid610 | 2015 | 4 |
| cid610 | 2016 | 12 |
| cid598 | 2013 | 4 |
| cid598 | 2014 | 4 |
| cid777 | 2014 | 35 |

# SQL Data Manipulation Operations

# Quiz

Consider the following two tables

Instructor

| ID | Name | Office |
|---|---|---|
| 4 | Jennifer | B308 |
| 35 | Paul | B311 |
| 12 | Kim | E112 |

Course

| CourseID | Year | Instructor |
|---|---|---|
| cid444 | 2012 | 35 |
| cid598 | 2013 | 4 |
| cid444 | 2013 | 35 |

What is the result of executing the following SQL statement?

**DELETE FROM** Instructor **WHERE** Name **LIKE** "%i%";

1) Removal of the *Kim* tuple from the *Instructor* table
2) Removal of both the *Kim* tuple and the *Jennifer* tuple from the *Instructor* table
3) an error message
4) it depends

# SQL Views

# What are Views?

- A **virtual** table **derived** from other (possibly virtual) tables, i.e. always up-to-date

```
CREATE VIEW dept_view AS
 SELECT DNO, COUNT(*) AS C, AVG(SALARY) AS S
  FROM    EMPLOYEE
  GROUP BY DNO;
```

- Example of usage in queries:

```
SELECT DNO  FROM dept_view  WHERE S > 25000 ;
```

- Why?
  - Simplify query commands
  - Provide data security
  - Enhance programming productivity

www.liu.se

LINKÖPING UNIVERSITY